Architecture and Design Methodology for Autonomic Systems-on-Chip (ASoC)

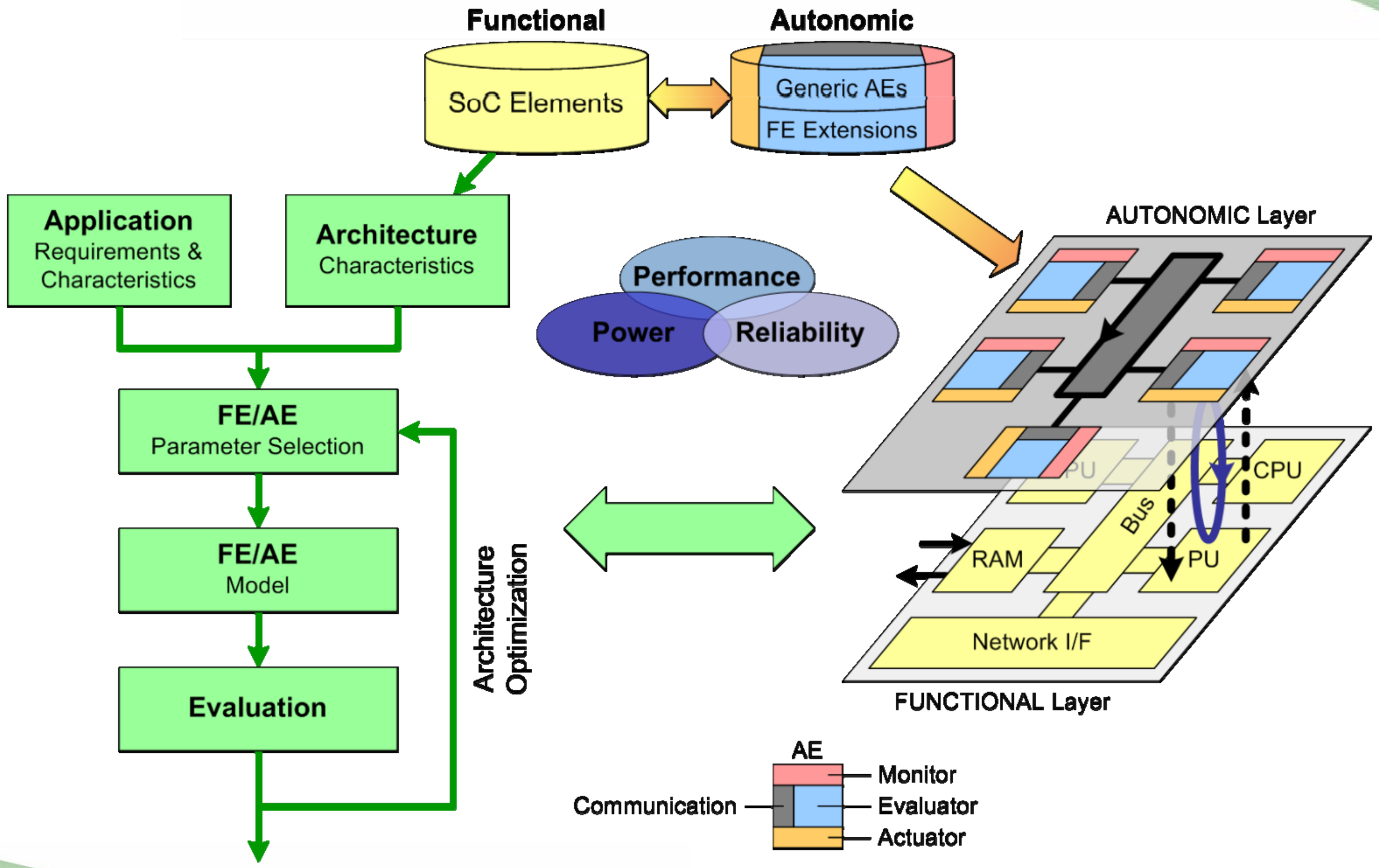
Johannes Zeppenfeld[1], Andreas Bernauer[2], Abdelmajid Bouajila[1],
Andreas Herkersdorf[1], Wolfgang Rosenstiel[2,3], Walter Stechele[1],
Oliver Bringmann[3]

[1]Technische Universität München

[2]Universität Tübingen

[3]Forschungszentrum Informatik

Project Reminder



Functional

SoC Elements

Autonomic

Generic AEs

FE Extensions

Application
Requirements & Characteristics

Architecture
Characteristics

Performance

Power

Reliability

FE/AE
Parameter Selection

FE/AE
Model

Evaluation

Architecture Optimization

AUTONOMIC Layer

CPU

PU

Bus

RAM

PU

Network I/F

FUNCTIONAL Layer

AE

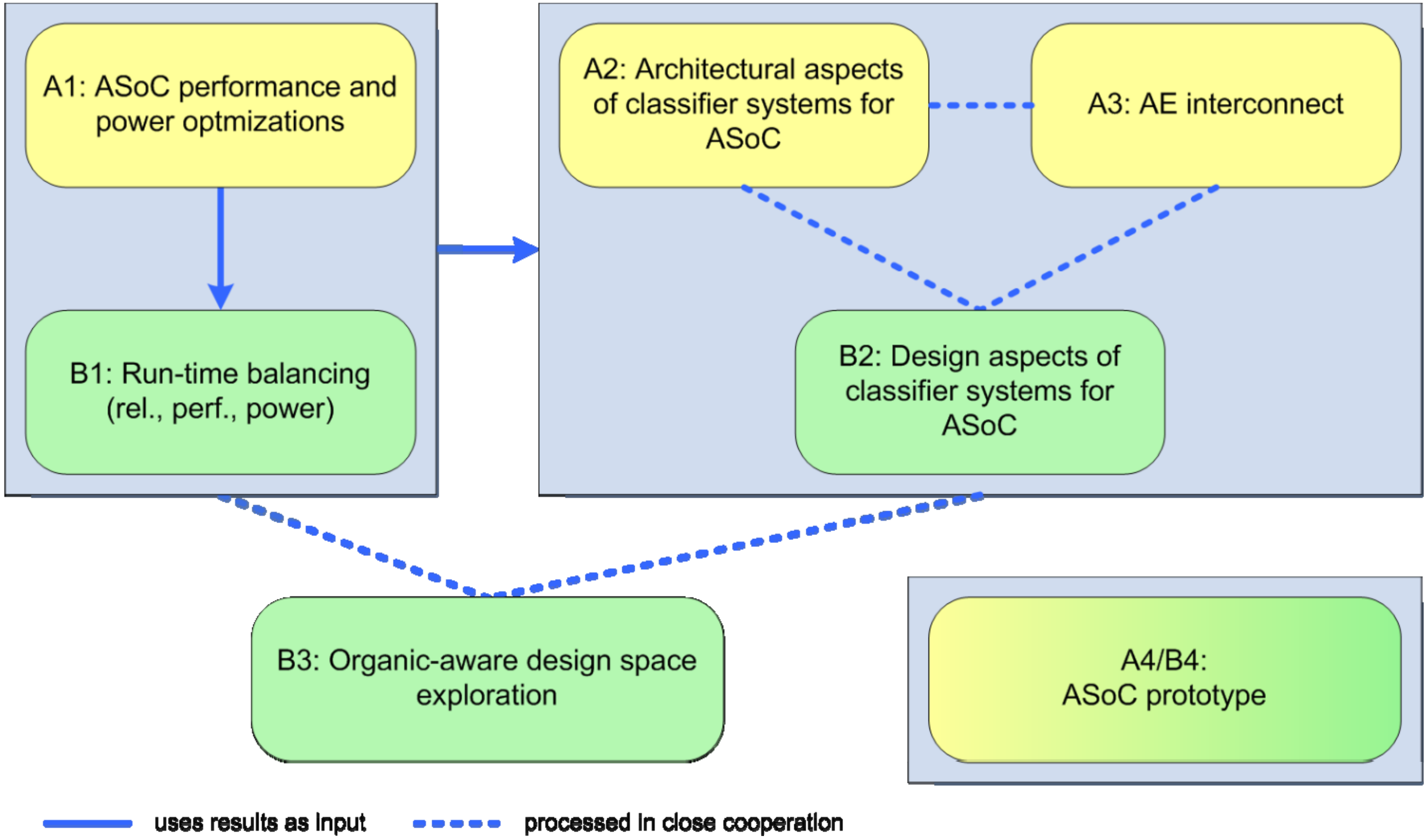
Communication

Monitor

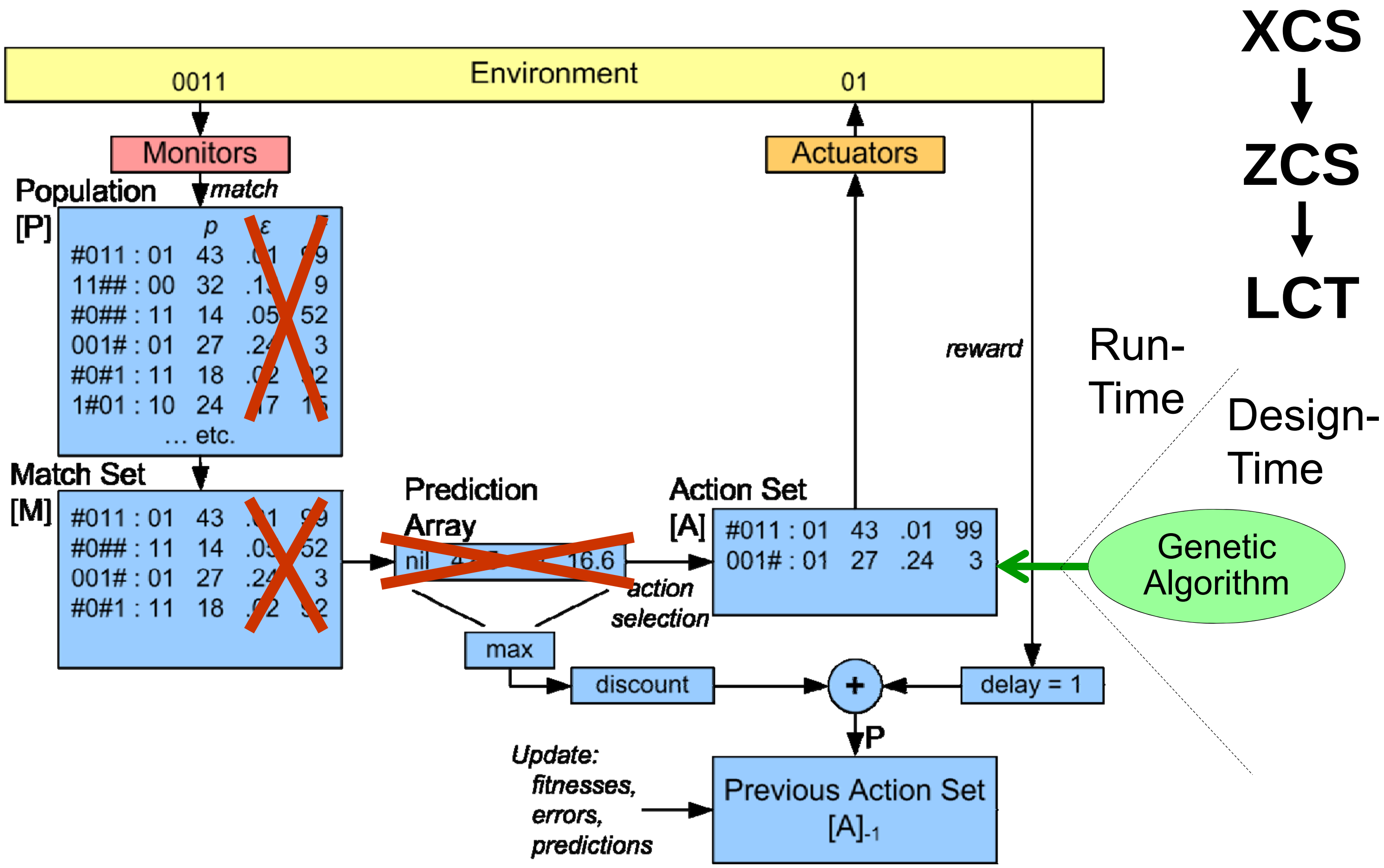
Evaluator

Actuator

Phase 2 Work Packages


A1: ASoC performance and power optmizations
B1: Run-time balancing (rel., perf., power)
A2: Architectural aspects of classifier systems for ASoC
A3: AE interconnect
B2: Design aspects of classifier systems for ASoC
B3: Organic-aware design space exploration
A4/B4: ASoC prototype
uses results as input
processed in close cooperation

Classifier Systems for ASoC



XCS
↓
ZCS
↓
LCT

Run-Time

Design-Time

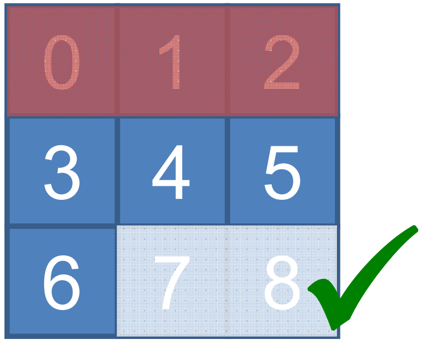
[Wilson95, Zeppenfeld08, Bernauer08a]

/ and SW at Design and Run Time

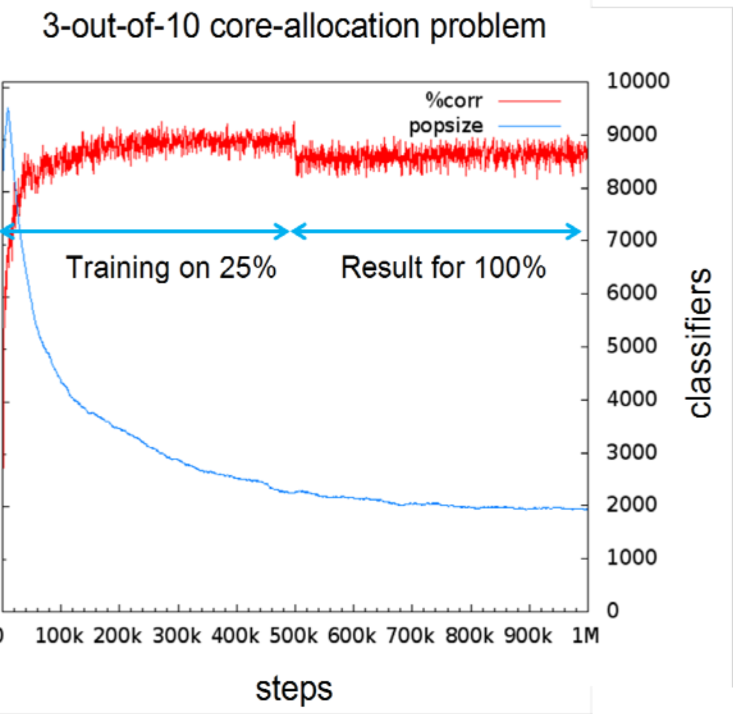
	Hardware	Software
sign me	• No hardware available • Simulation based only • Previous generations of an autonomic system can be used to collect data for further design cycles	• Full XCS implementation for best possible learning • Design tools such as ASoCsim running offline • Generation of rules to be used at run time • Determination of parameters to be used in system
un me	• Learning classifier table as hw-optimized reinforcement learning algorithm optimized for hardware • Learning through dynamic adjustment of fitness value as indication of rule effectiveness • Organic self-x properties can be	• Autonomic configuration software running on system • Long term goal specification through parameterization of target function • Responsible for rule replacement in

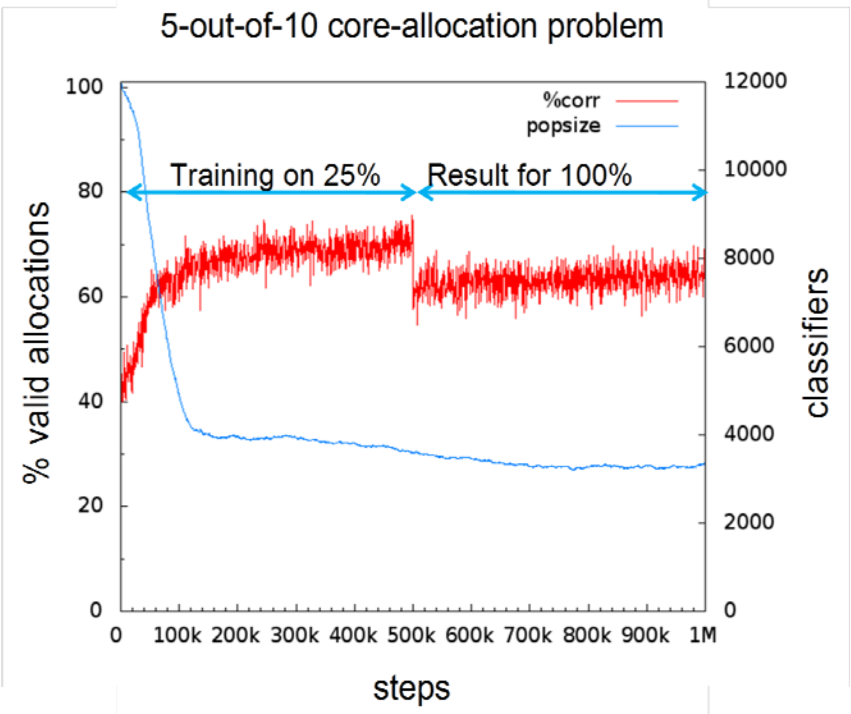
sign Example: Core Allocation

- iven c cores, partially occupied, elect n free cores
- xample: c=9, n=2
  1. Random number of cores are occupied: Input (condition): 111000000
  2. Evaluator chooses allocation by index: Output (action): 000000011
  3. Evaluator receives reward for a valid allocation (no allocated cores were occupied)
- djust difficulty by putting constraints n free cores

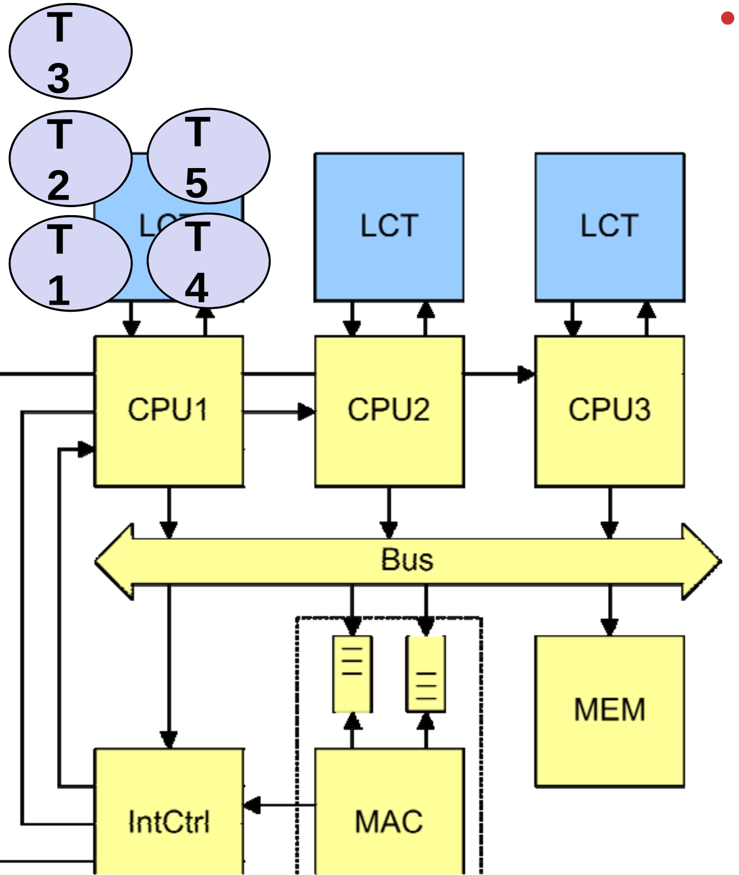


aining During Design Time





ntime Example: MP-SoC



- Applications, Objectives, Rewards
  - Self-organized workload balancing among multiple CPUs
  - Objective function to minimize:

$$\delta_{Load} = |f_{cpu\ n} \cdot util_{cpu\ n} - f_{cpu\ avg} \cdot util_{cpu\ avg}|$$

$$\delta_{Util} = 1.0 - util_{cpu\ n}$$

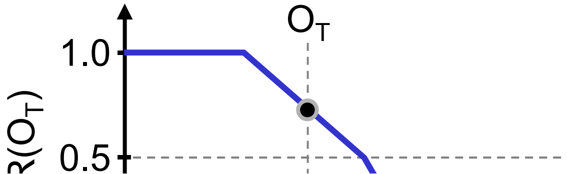
$$\delta_{Freq} = f_{cpu\ n}$$

$$O_{CPU} = w_1 \cdot \delta_{Load} + w_2 \cdot \delta_{Util} + w_3 \cdot \delta_{Freq}$$

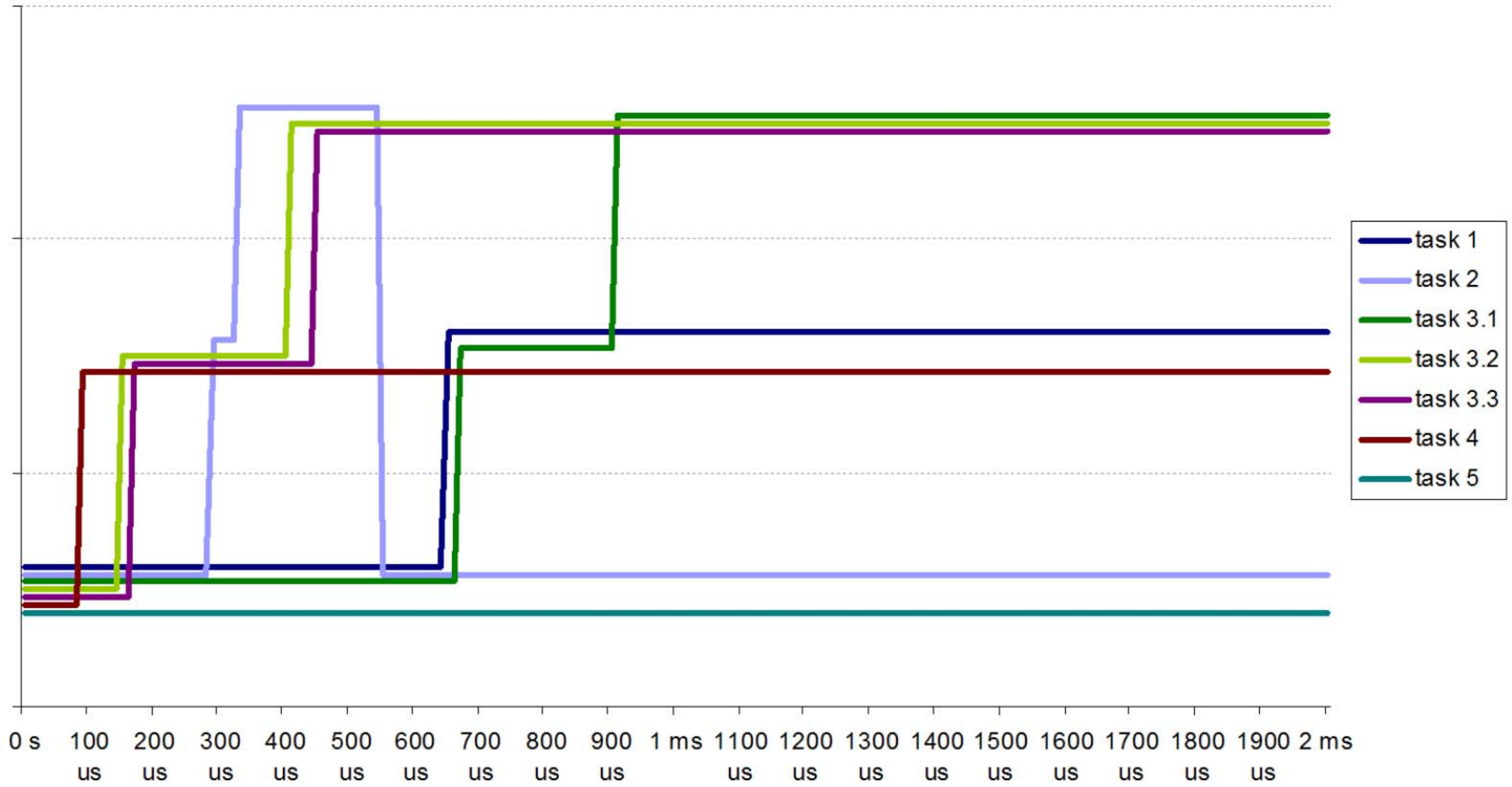
$$O_{Sys} = w_a \cdot O_{CPU1} + w_b \cdot O_{CPU2} + \ldots$$

$$O_T = O_{Sys} \text{ at time } T$$

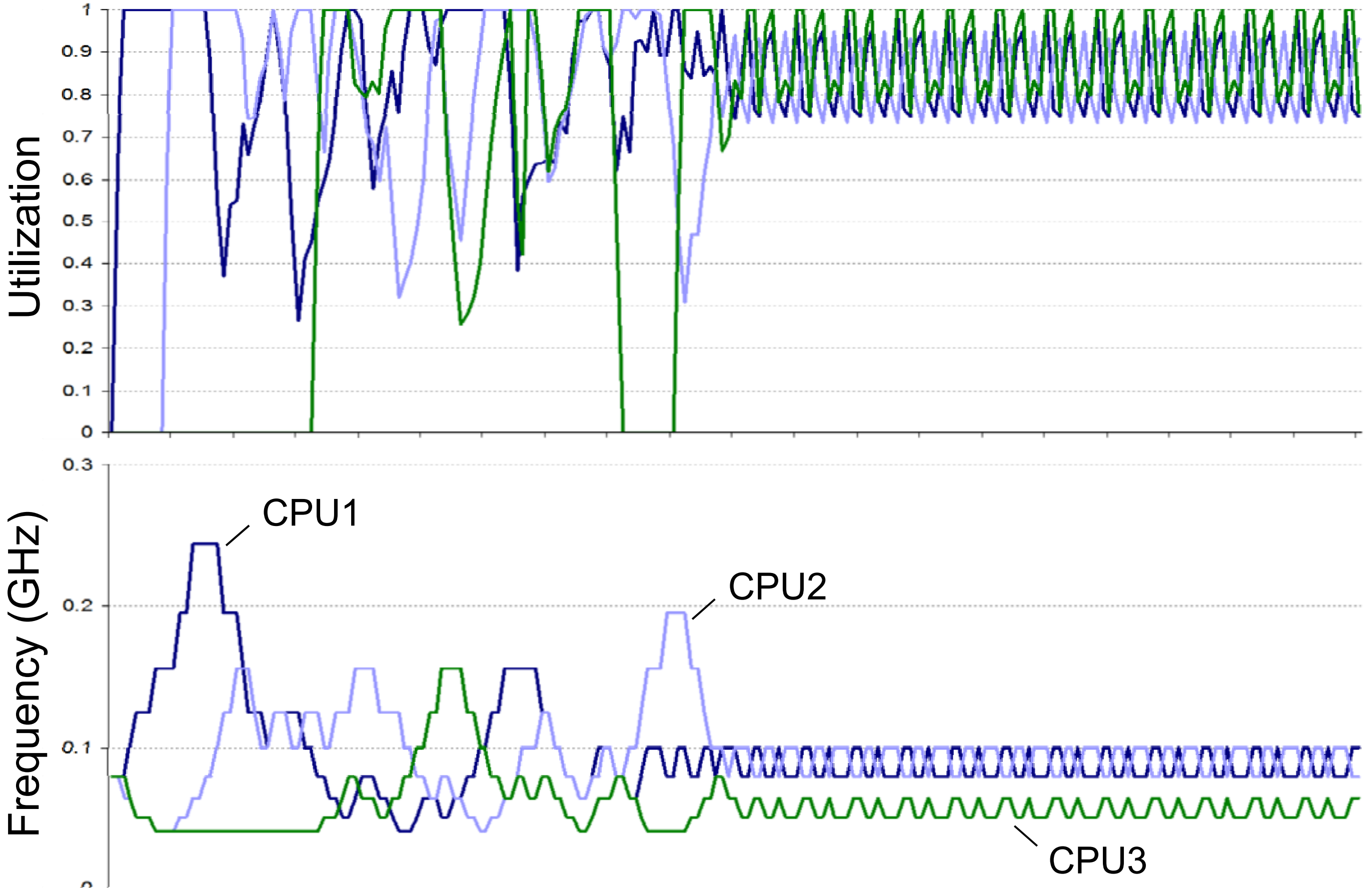
  - Reward function:



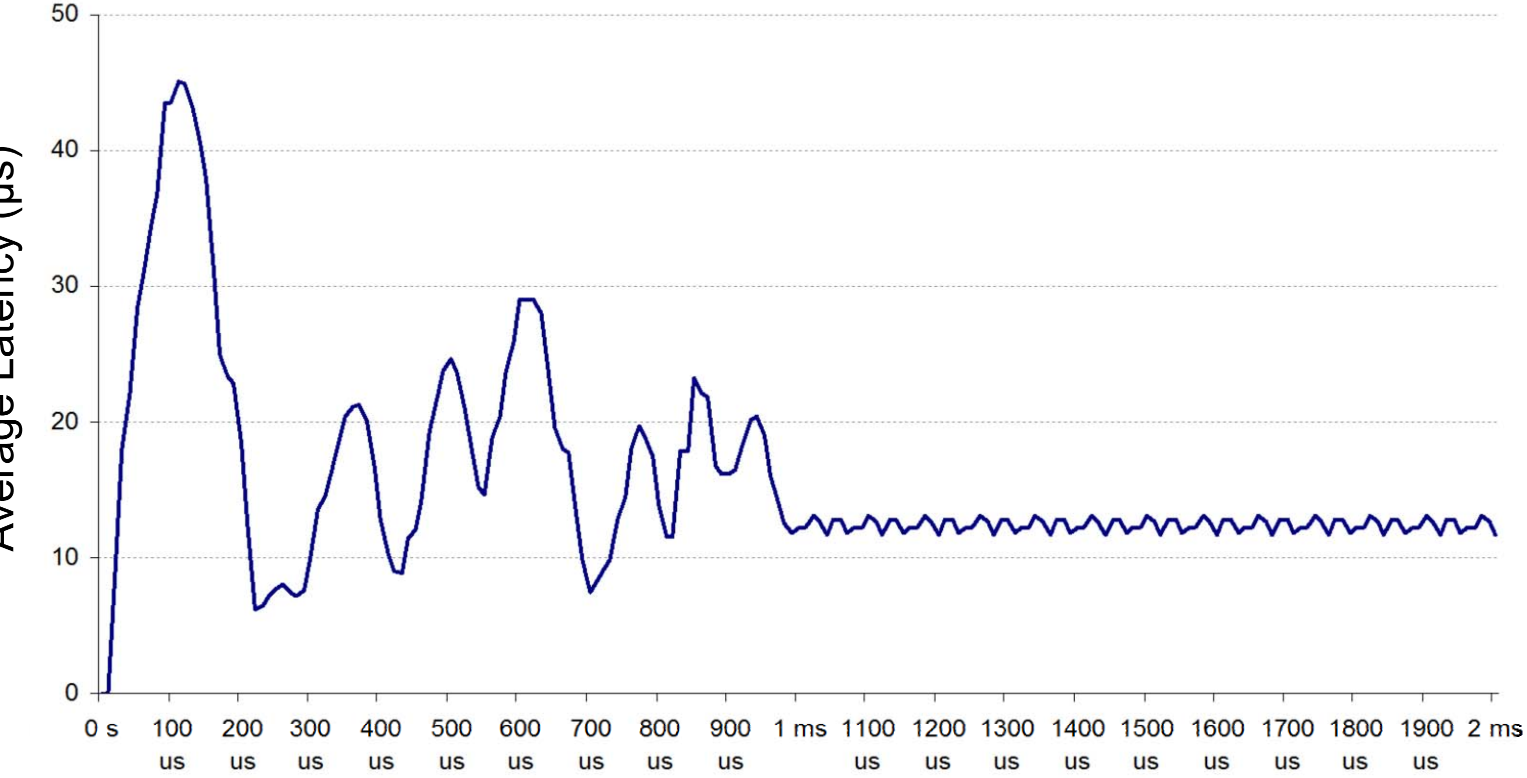
tonomic Task Migration



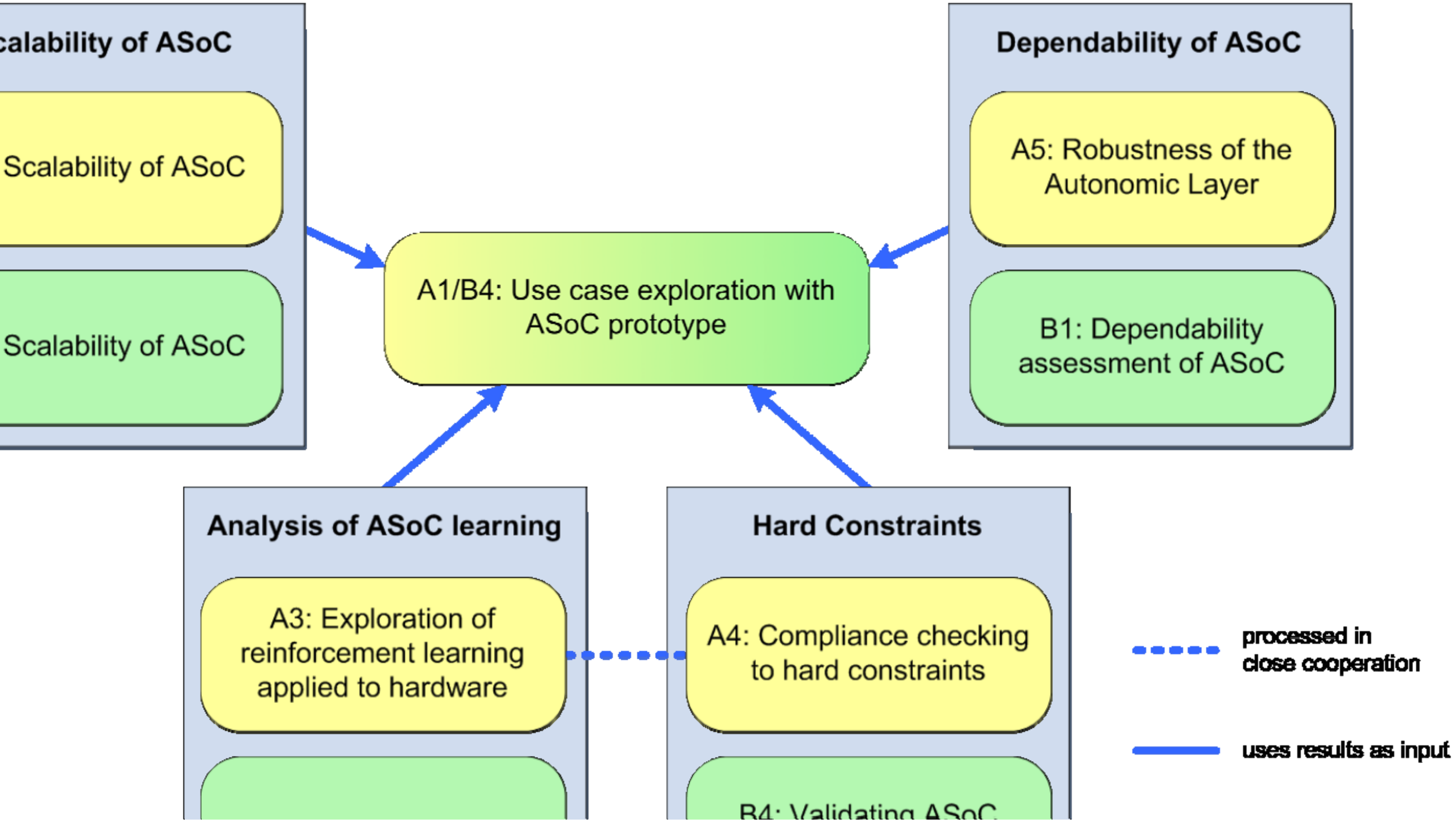
tonomic Frequency Adjustment



sulting Packet Latency



ase 3 Work Packages



calability of ASoC

Scalability of ASoC

Scalability of ASoC

A1/B4: Use case exploration with ASoC prototype

Dependability of ASoC

A5: Robustness of the Autonomic Layer

B1: Dependability assessment of ASoC

Analysis of ASoC learning

A3: Exploration of reinforcement learning applied to hardware

Hard Constraints

A4: Compliance checking to hard constraints

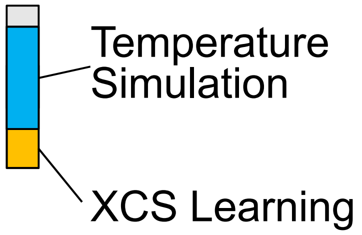
B4: Validating ASoC

processed in close cooperation

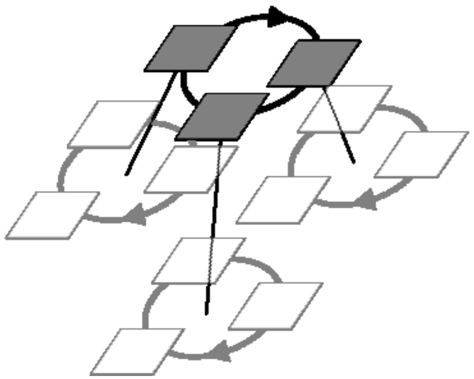
uses results as input

alability of ASoC

- ow to efficiently train XCS for systems with many cores
  - Reduce simulation time due to temperature estimation
  - Reduce simulation time due to rule table generation
- /hat is the effect of many distributed evaluators (LCTs)



Reduce simulation time



Hierarchical AE interconnect

pendability Assessment

- ecision system should be able to handle failures in all omponents (in AE as well as FE)
- ependability of resulting design should be assessable

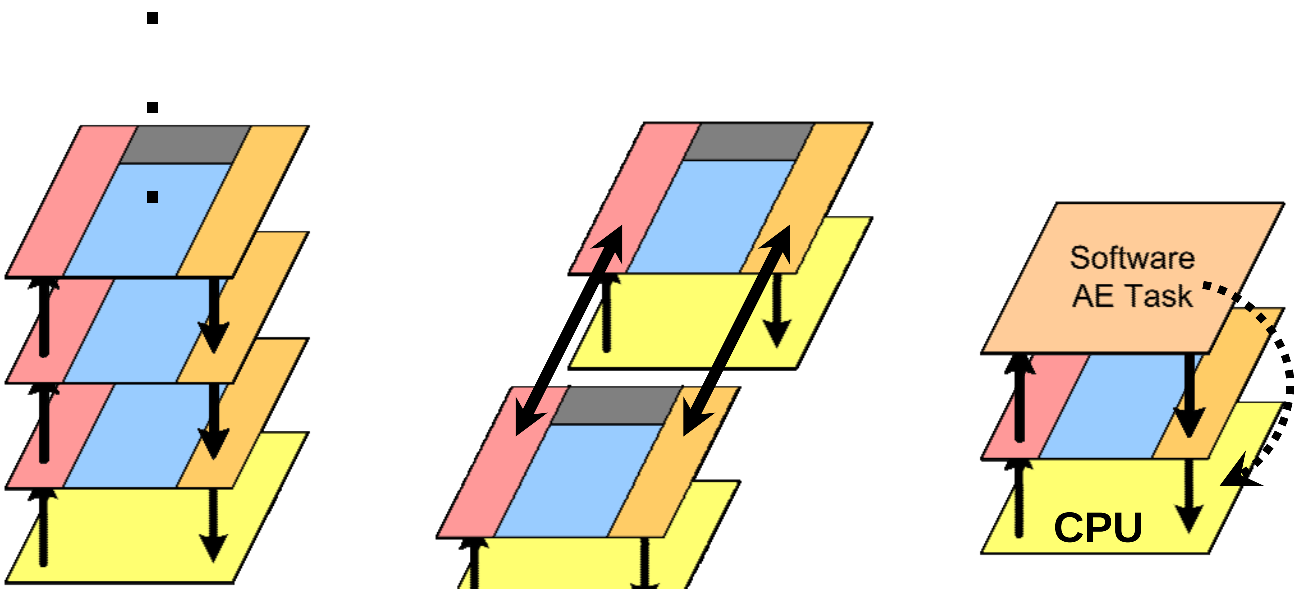
- rain decision system using TLM fault injection
  - Continue injecting errors on the functional layer
  - Add error injection to the autonomic layer
- igh-level simulation model of failures at all locations
  - Model failures of independent components
  - Model failures of dependent components
  - Classify errors (no error, wrong output, deadlock, …)
- ssess resulting ASoC dependability

bustness of the Autonomic Layer

lonitors to monitor the autonomic layer

Es will supervise both FE and neighboring AEs

oftware monitoring task verifies AE functionality



sts of Emergence

heoretical analysis of XCS

- Minimum number of classifiers?
- Necessary duration of learning at design time?
- Upper bounds on duration of self-optimization at run time?
- Necessary rule-update frequency at run time?

valuate two further rule-based decision systems

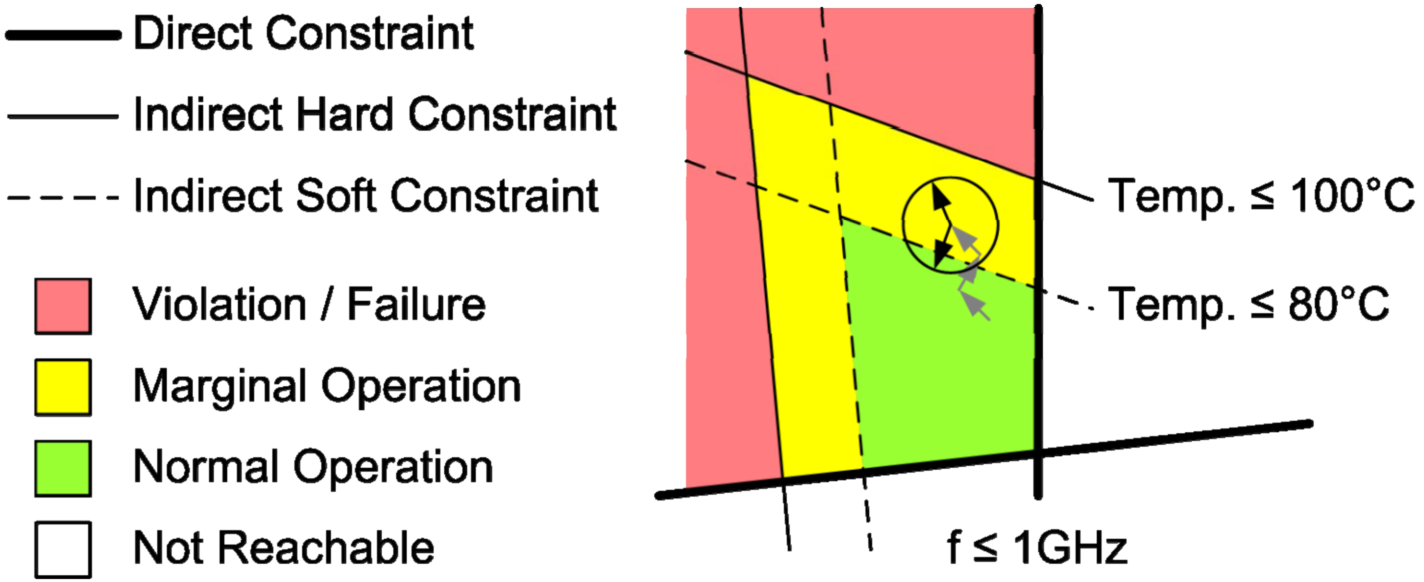
- ARON (in cooperation with Prof. Brockmann)
- X-TCS (semi-Markov to solve timing problem)

reliminary results

- Predict number of necessary classifiers and learning time
- Level of knowledge of the goal has significant impact on complexity

tisfying Hard Constraints

- iolating soft constraints leads to performance egradation.
- iolating hard constraints could lead to system failure.



▬ Direct Constraint

— Indirect Hard Constraint

- - - - Indirect Soft Constraint

🟥 Violation / Failure

🟨 Marginal Operation

🟩 Normal Operation

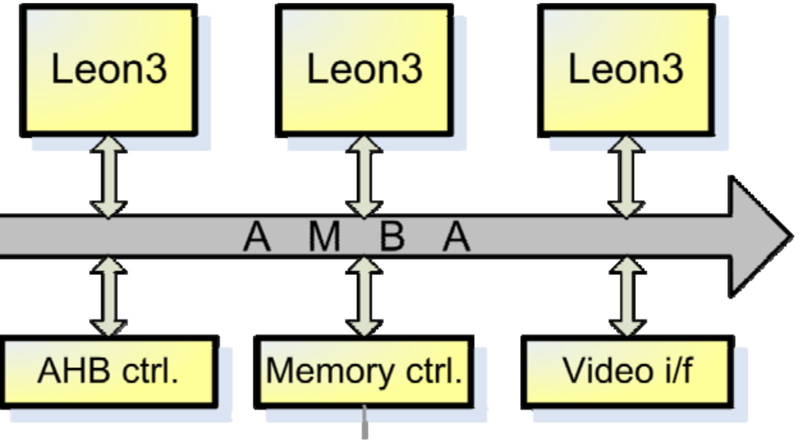
□ Not Reachable

oC Prototype

eal-world applications running on Leon3-based
rototype:

- Networking (Varying packet rate, type and size)
- Video Processing (Video frames > 300 kB @ 25 frames/sec)

erify that the ASoC FPGA prototype performs self-
ptimization, self-correction and learning.



operations

- eam of Prof. Reif
  - Verification of self-x properties based on logic model
  - Tools for reliability estimations
- eams of Prof. Müller-chloer / Prof. Schmeck
  - LCS concept and implementation
- eam of Prof. Maehle / rof. Brockmann
  - Cooperation on HW/SW

- Team of Prof. Ernst
  - Interested in reliability estimation
- DodOrg Karlsruhe
  - Organic Middleware

ase 2 Publications

ernauer08a] A Bernauer, D Fritz, W Rosenstiel, *Evaluation of the Learning Classifier System XCS for SoC run-time ntrol* , Lecture Notes in Informatics, Vol. 134, p.761-768, Springer, Gesellschaft für Informatik

ernauer08b] A Bernauer, D Fritz, B Sander, O Bringmann, W Rosenstiel, *Current state of ASoC design thodology*, http://drops.dagstuhl.de/opus/frontdoor.php?source_opus=1564, ISSN 1862-4405

ernauer09] A. Bernauer, W. Rosenstiel, O. Bringmann, *Generic Self-Adaptation to Reduce Design Effort for stem-on-Chip*, SASO 2009

erkersdorf08] A Herkersdorf, J Zeppenfeld, A Bouajila, W Stechele, *Hardware-Supported Learning Classifier bles in Autonomic Systems on Chip*, Dagstuhl Seminar 08141, March 30 - April 4, 2008

nkes07] A Lankes, T Wild, J Zeppenfeld, *Power estimation of Variant SoCs with TAPES.* In: Euromicro- DSD 07

ehl09] A. Viehl, B. Sander, O. Bringmann, W. Rosenstiel, *Analysis of Non-functional Properties of MPSoC signs*, Languages for Embedded Systems and their Applications, Lecture Notes in Electrical Engineering, Vol 36

ppenfeld08] J Zeppenfeld, A Bouajila, W Stechele, A Herkersdorf, *Learning Classifier Tables for Autonomic stems on Chip,* Lecture Notes in Informatics, Vol. 134, p.769-776, Springer, Gesellschaft für Informatik

ppenfeld10sub] J. Zeppenfeld, A. Bouajila, W. Stechele, A. Herkersdorf, *Autonomic Workload Balancing for lticore Processor Systems*, submitted to ARCS 2010

blications and References

er08a] A Bernauer, D Fritz, W Rosenstiel, *Evaluation of the*
g Classifier System XCS for SoC run-time control , Lecture Notes
natics, Vol. 134, p.761-768, Springer, Gesellschaft für Informatik

er08b] A Bernauer, D Fritz, B Sander,
mann, W Rosenstiel, *Current state of ASoC design methodology*,
ops.dagstuhl.de/opus/
or.php?source_opus=1564, ISSN 1862-4405

er09] A. Bernauer, W. Rosenstiel, O. Bringmann, *Generic Self-*
ion to Reduce Design Effort for System-on-Chip, SASO 2009

sdorf08] A Herkersdorf, J Zeppenfeld,
ila, W Stechele, *Hardware-Supported Learning Classifier Tables*
nomic Systems on Chip, Dagstuhl Seminar 08141, March 30 -
2008

07] A Lankes, T Wild, J Zeppenfeld, *Power estimation of Variant*
ith TAPES. In: Euromicro- DSD 2007.

9] A. Viehl, B. Sander, O. Bringmann, W. Rosenstiel, *Analysis of*
ctional Properties of MPSoC Designs, Languages for Embedded
s and their Applications, Lecture Notes in Electrical Engineering,

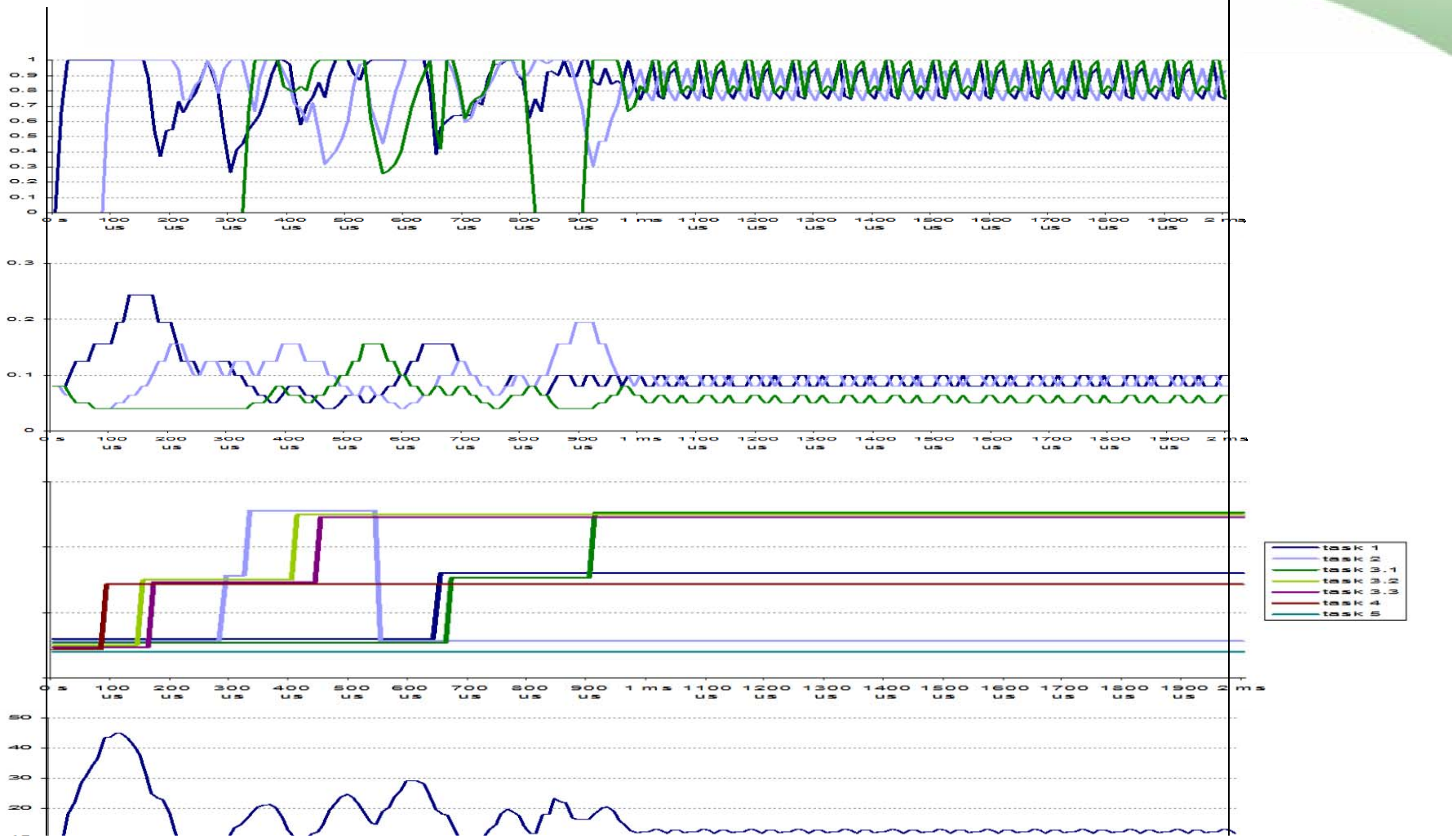
nfeld08] J Zeppenfeld, A Bouajila, W Stechele, A Herkersdorf,
g Classifier Tables for Autonomic Systems on Chip, Lecture Notes
natics, Vol. 134, p.769-776, Springer, Gesellschaft für Informatik

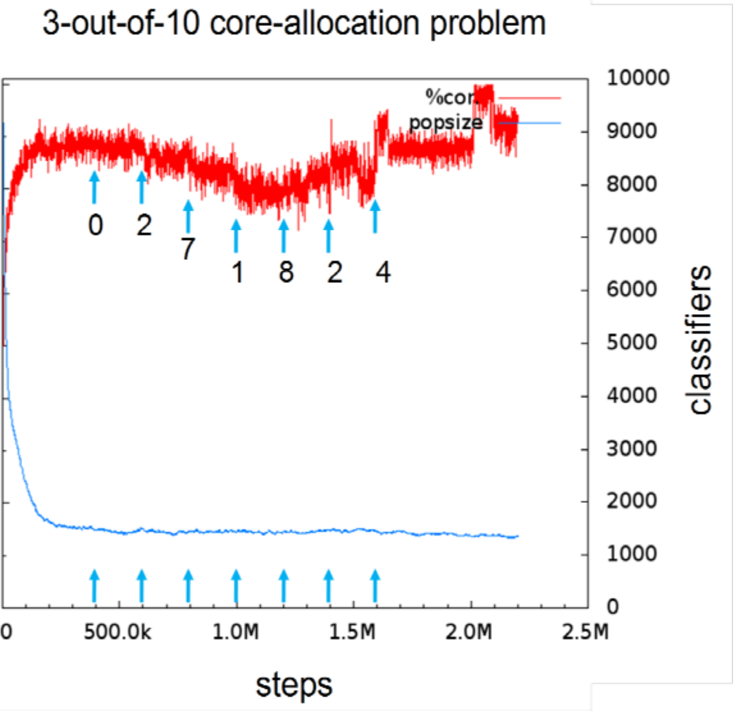
nfeld10sub] J. Zeppenfeld, A. Bouajila, W. Stechele, A.
dorf, “Autonomic Workload Balancing for Multicore Processor
s”, submitted to ARCS 2010

- [Gold03] A Gold, A Kos, *Temperature Influence on Power Consumption and Time Dealy*, Dep. Electronics, Cracow, Poland, 2003.
- [Huang04] W Huang, MR Stan, K Skadron, K Sankaranarayanan, S Ghosh, S Velusamy, *Compact Thermal Modeling for Temperature-Aware Design,* DAC 04.
- [Wilson95] S. Wilson, *Classifier fitness based on accuracy*, Evolutionary Computation, 3, 1995, pp. 149-175
- [Zhu06] D Zhu, *Reliability-Aware Dynamic Energy Management in Dependable Embedded Real-Time Systems,* RTAS'06, IEEE Computer Society, 2006, p. 397-407

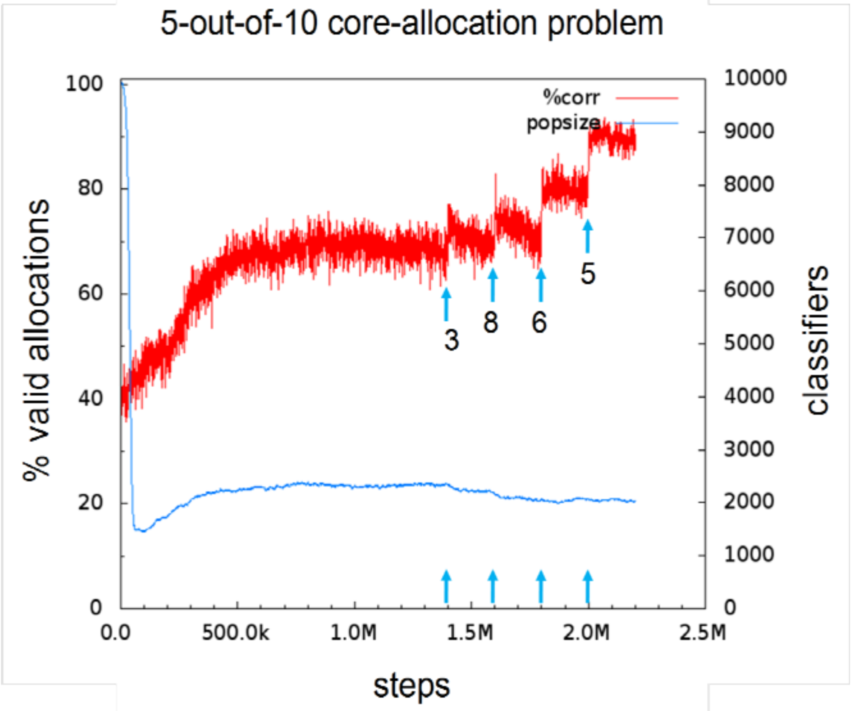
ckup

sulting Packet Latency





3-out-of-10 core-allocation problem



5-out-of-10 core-allocation problem

Interconnect

rovides an interconnect between the AEs in order to ermit IPs status exchange and therefore global ptimization

E interconnect:

- AE interconnect is independent from FE interconnect
- Serial ring architecture
  - Simple in comparison to other buses (AMBA, PLB..)
  - Suffices requirements (bandwidth, latency)
  - Optimized version of serial ring (add control bits in order to simplify logic and save registers)

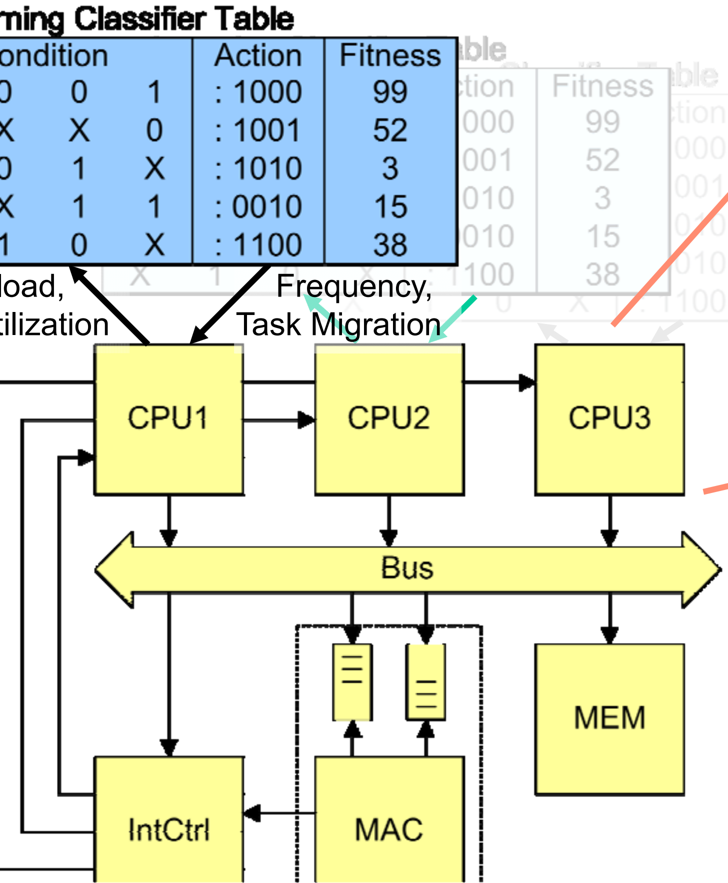
nplementation results

	2 bit ctrl + 1 bit data	2 bit ctrl + 4 bit data
esources	246 slices	300 slices

	Hardware	Software
me	□ No hardware available □ Simulation based only □ Previous generations of an autonomic system can be used to collect data for further design cycles	□ Design tools such as ASoCsim running offline □ Full XCS implementation for best possible learning capabilities □ Generation of rules to be used at run time □ Determination of parameters to be used in system (number of cores, protection scheme, size of classifier population, etc.)
ne	□ Learning classifier table as reinforcement learning algorithm optimized for hardware □ Learning through dynamic adjustment of fitness value as indication of rule effectiveness □ Low level monitoring of system parameters and immediate intervention to prevent critical faults or error propagation □ Organic self-x properties can be validated with an operational ASoC FPGA prototype	□ Autonomic configuration software running on the system □ Long term goal specification through parameterization of target function (global RPP metric) □ Responsible for rule replacement in LCT hardware □ Preparation of assorted monitor signals for a potential autonomic middleware (not in the scope of ASoC)

elf-Organizing MPSoC

[Zeppenfeld08]



ning Classifier Table

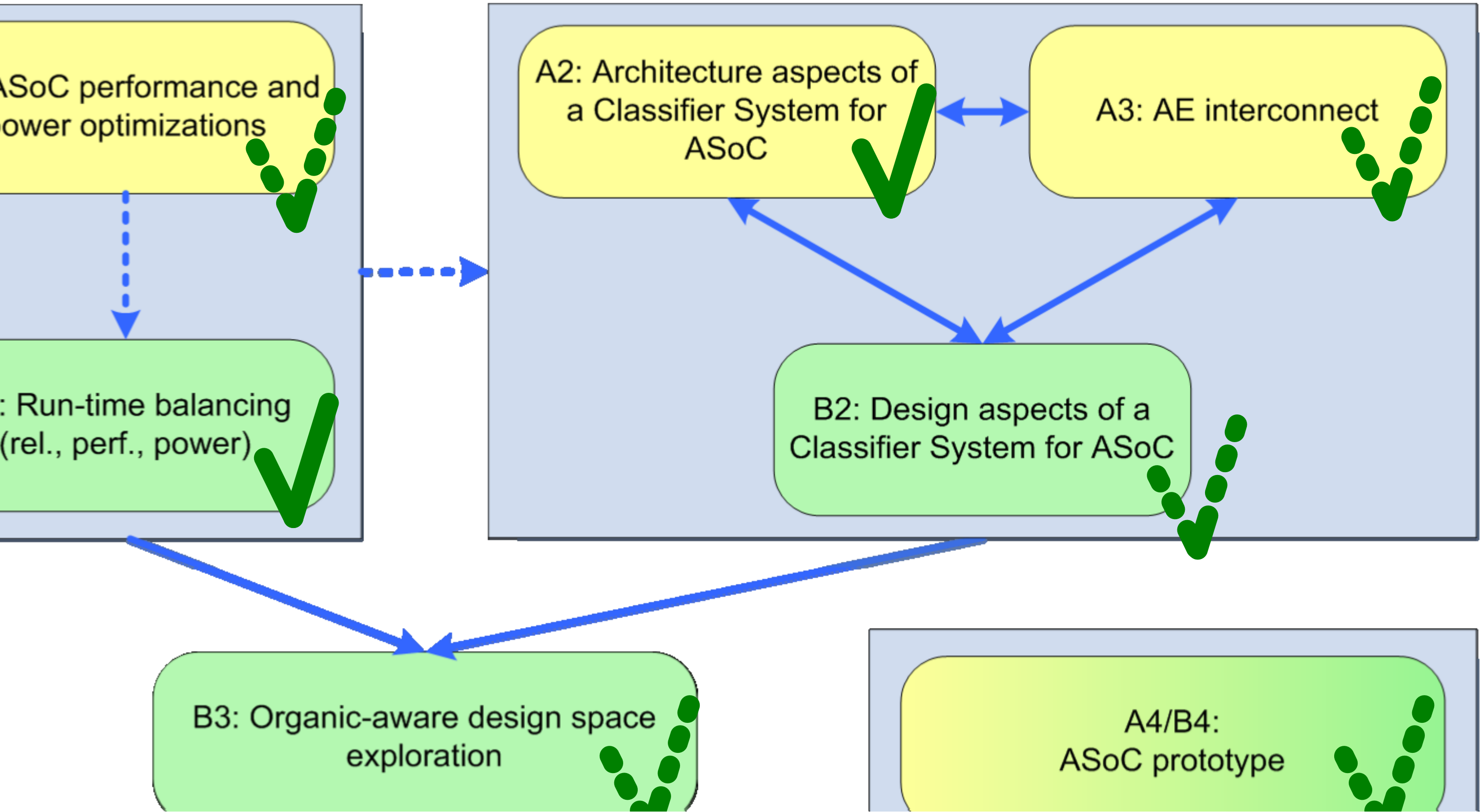
ondition			Action	Fitness
0	0	1	: 1000	99
X	X	0	: 1001	52
0	1	X	: 1010	3
X	1	1	: 0010	15
1	0	X	: 1100	38

```
If (CPU util high AND f is high) then
   Migrate task
Else if (CPU util high AND f is low) then
   Increase f
Else if (CPU util low AND f is high) then
   Decrease f
End
```

Learning Classifier Table:

- HW implementation of XCS-based reinforcement machine learning technique [Wilson95]
- Multi-conditional rules with actions and reward oriented fitness evaluation

mmary



ASoC performance and ower optimizations ✓

: Run-time balancing (rel., perf., power) ✓

A2: Architecture aspects of a Classifier System for ASoC ✓

A3: AE interconnect ✓

B2: Design aspects of a Classifier System for ASoC ✓

B3: Organic-aware design space exploration ✓

A4/B4: ASoC prototype ✓